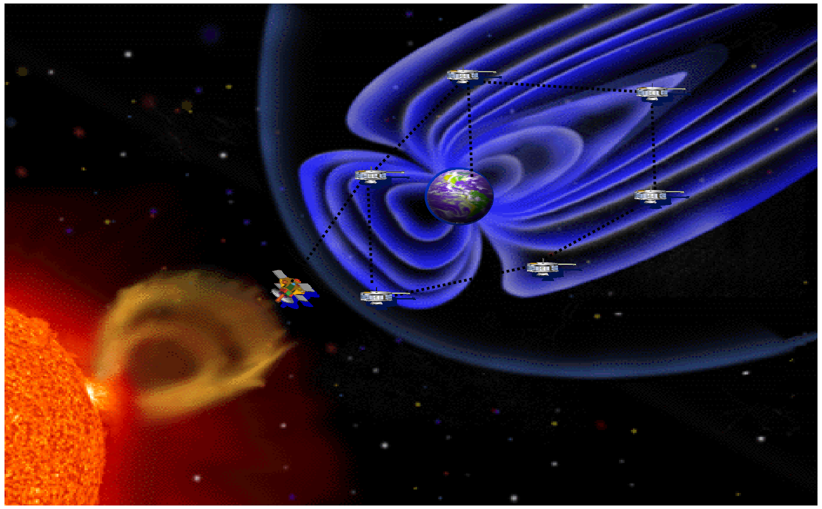
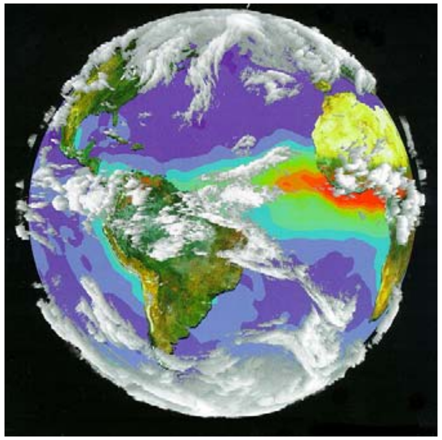
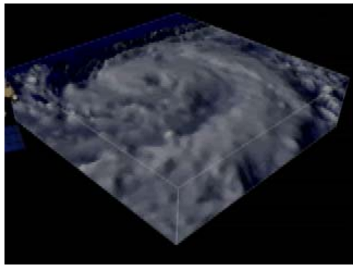
Information Systems Division / 580

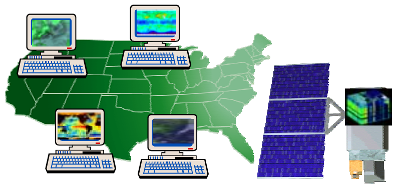


Advanced flight and scientific information systems will support the execution and analysis of the scientific measurements and observations of the Earth and the Sun-Earth system.



Good Software Practices





Interoperable Models

Dec. 2002

Session Goal

To provide a general awareness of <u>software management issues</u> for Goddard Technical Managers who want to gain some familiarity with software issues that may affect the success of their projects.

GSFC Mission Operations Software

Basic Definitions

- Software:
  - Programs, procedures, rules, and any associated documentation pertaining to the operation of a computer system [ANSI/IEEE Standard 729-1983].
- Software Engineering:
  - The systematic approach to the development, operation, maintenance, and retirement of software through the use of suitable standards, methods, tools, and procedures [ANSI/IEEE Standard 729-1983].
    - Encompasses many diverse activities including requirements analysis, architectural and detailed design, implementation (coding, programming), assurance, testing, and maintenance.
    - <u>Not</u> another term for programming or coding.
- <u>Software Management:</u>
  - A disciplined approach to the planning, tracking, assessing, and controlling of software product development through the selection and use of specific methods, tools, and procedures [JPL D-2352].

Trends in GSFC Software

- Hardware memory and processors have increased significantly in capacity and speed, but more is needed. MAP used 260MB of memory, for example.
- Software applications have become larger and more complex.
- More autonomy in the spacecraft and more complex instruments and data processing.
- Much more software, both on-board and in the ground system.
- Ground and flight software, in concert, is assuming an increasing role in mission capability and performance.
- Software costs can be as high as 25% of project costs.
- Greater use of generalized hardware with customization in the software (and a desire to move toward reusable software components).
- System architecture is an integrated hardware-software design.

Software is an integral and critical part of today's systems.

Perceptions -- Two Sides of the Software Coin

Views of Project Managers (PMs)

1. SWEs are not good at estimating costs and schedules.

2. It's hard to know the true status of the software. I can't tell what' going on "over there"!

3. Why does it "take so long" to develop software?! Isn't it "just a small matter of programming"?

Views of Software Engineers (SWEs)

1. PMs don't accept realistic estimates, but insist on cuts OR they come to us with already approved budgets and schedules ("done deal") that are unrealistic.

2. PMs often don't take the time or know how to ask the right questions to get to the heart of issues OR they ignore our warnings about the effects of tradeoffs.

3. PMs don't appreciate the difficulty of software or understand the "ripple effect" of changes.

And to Strike Fear in the Heart …

- The often cited Standish Group 1998 Study on large software projects in the late 1990s reported
  - 53% either delivered late or exceeded budget
  - 31% were cancelled
  - 16% were successful

- Average cost for a "failed" project is 189% of the original estimate
- Average delivery for a "failed" project is 222% of the original schedule
- On average "challenged" projects deliver 61% of specified functions
- The Standish Group recently performed a survey of IT managers:
  - 27% believe there are now significantly more software failures
  - 21% believe there are somewhat more failures
  - 11% believe things haven't changed at all
  - 19% believe there are somewhat fewer failures
  - 22% believe there are significantly fewer failures

  Only 41% believe there are fewer failures now than five years ago !

GSFC Mission Operations Software

General Mission Software Context

Mission Software Good Basic Practices

Recent GSFC Software Problems

Steps to Successful Software Products

Improvements In-Process/-Plan

Summary

Mission Information Systems Providers

End-to-end data systems engineering of mission systems

Embedded spacecraft, instrument and hardware component software

Real-time ground mission data systems for spacecraft integration and on-orbit ops (e.g., S/C command & control, launch and tracking services)

Science data systems including data processing, archival, distribution, analysis & info mgmt.

Off-line mission data systems (e.g., command mgmt., S/C mission and science planning & scheduling, guidance & navigation, network scheduling)

Advanced concept development for archival, retrieval, display, dissemination of science data

Technology R&D focused on autonomy, scientific analysis tools, distributed computing architectures
and
Tools and services in support of information management

ISD: End-to-End Information Systems Providers

Branch	Functional Area/Products	Services
581/Systems Integration & Engineering *Margaret Caulfield, Vacant ABHs*	End-to-end data systems engineering of ISD mission systems development activities.	Mission directors, ground sys/flight ops management, sys. eng., flight prep support, SW eng, Sys I&T, AO prep
582/Flight Software *Elaine Shell, Ray Whitley, Kequan Luu, Ron Zellar*	Embedded spacecraft, instrument and hardware component softwares; FSW testbeds	End-to-end FSW development; simulation s/w; spacecraft sustaining engineering
583/Mission Applications *Henry Murray, Scott Green*	Off-line mission data systems (e.g., Command man., s/c mission and science P&S, GN&C, NCC	Sys. eng.& implementation, COTS application, testbeds for concept proof/prototyping in ops environment
584/Real-Time Software Engineering *Vacant BH, Dwayne Morgan/WFF, John Donohue*	Real-time ground mission data systems for I&T and on-orbit ops (e.g., s/c command & control, launch and tracking services)	Sys. eng.& implementation, COTS application, simulators, testbeds for concept proof/prototyping in ops env.
585/Computing Environments & Technology *Howard Eiserike, Steve Naus*	Tools and services in support of information management	Network manage., business/support tool develop, WWW applications
586/Science Data Systems *Mike Seablom, Vacant ABH*	Science data systems including data processing, archival, distribution, analysis & info man.	Sys. eng.& implementation, COTS application & integration, testbeds, prototyping
587/Advanced Data Management and Analysis *Jim Byrnes*	Advanced concept development for archival, retrieval, display, dissemination of science data	Next-gen req. development, testbed for sys evaluation, prototype products
588/Advanced Architectures & Autonomy *Julie Breed, Barbie Brown-Medina*	Technology R&D focused on space-ground automation and autonomy sys, advanced architectures, and advanced scientific tools and systems	Sys. eng & implementation, human-computer eng., technology evaluations, concept prototypes, sw eng. methods

GSFC Mission Critical Software Implementation Overview (1 of 2)

- **System elements are often based on prior mission systems identified in Formulation up to Preliminary Design (a reuse & evolution approach)**

- **Core software practices follow generally accepted Software Engineering practices such as:**

  Documents
  - **Product Development Plan, Requirements, IRD/ICDs, Operations Concepts, …**

  Formal Reviews
  - **Requirements, Preliminary & Critical Design, Test Readiness, Mission Ops., …**

  Practices
  - **Design and code walkthroughs**
  - **Software development folders**
  - **Early and intermediate build and test**
  - **Requirements-to-test matrices**
  - **Test scenario development and walkthroughs**
  - **Test coverage and test results analysis and sign-off**
  - **Problem id and tracking**
  - **Configuration Management processes**

GSFC Mission Critical Software Implementation Overview (2 of 2)

- Selected implementation methodologies are Project & domain specific, ranging from Yourdon methods to Object Oriented (OO) methods, using waterfall to spiral approaches, and implemented utilizing assembly computer languages to Java

- New technologies and software engineering practices are adapted when benefits/risks are accepted by Projects, e.g.
  - Health & Safety (H&S) and routine command operations automation in IMAGE, XTE, TRMM, etc., enabling lights-dim operations cost savings
  - adaptation of LabView for ISTP science level 0 data processing to prolong the ISTP system life by reducing operations costs
  - Auto code generation in the MAP, SDO, GPM attitude control systems
  - Rational Rose Real-time autocode generation for JWST Common flight software

- Critical systems are evaluated by the NASA IV&V Facility for process integrity … covering requirements/design/code/test

GSFC's reuse-and-evolve approach is a strategy that is ' cost effective' and risk averse, while enabling tailoring to specific needs and accumulating functionality

Different Domains of Software Each Reflect A Different Emphasis

Flight Software
- driven by limited S/C life, asset survival, & mission science program
- continuous critical real-time ops, from attitude control to H&S monitoring
- fixed & constrained environment
- minimize risk with a never fail mindset
- restricted maintenance opportunities

Mission Control Ground Systems
- driven by limited S/C life, asset health, and observatory user demands
- episodic real-time & near-time ops, from command uplink to system state evaluations
- open to needs based augmentation
- risk adverse with a fail soft/over mindset
- full shadow maintenance capability

Science Data Management & Data Processing
- data retention & integrity driven
- near-time and later ops, from raw archival to signature calibrations and analysis
- flexible & extendable environment
- data fail soft mindset
- shadow mode and add-on maintenance

Science Data Dissemination
- science evolution & user driven
- near-time and later ops
- large user communities
- evolving user interfaces & access demands
- timely data delivery mindset
- shadow mode and add-on maintenance

… Domain Tailored Development & Qualification Approaches

Broad Range of Domain Expertise

- GSFC mission success is built upon long term staff experience across a broad range of missions in earth and space science

- Significant enablers include close interaction with the GSFC science community, S/C hardware engineering, and mission flight operations personnel

- Specific software application domains include:
  - S/C and instrument flight software
  - ground command & control systems, including ground stations
  - guidance & navigation operations systems
  - science and mission planning & scheduling systems
  - science data processing, archival, distribution, and calibration systems
  - science data analysis & modeling systems

- Software definition, development, and qualification is tailored to each Project with risk mitigation strategies keyed to mission criticality and software domain

Software Life-Cycle Products

Basic Software life-cycle "products" include

- **Software Product Plan, Software Schedule with Dependencies**
- **Software Requirements Document (SRD)**
- **Preliminary & Critical Design Reviews**
- **Release/build Plan**
- **Interface Control Document(s) (ICDs)**
- **Information/Data Needs Schedules**
- **Functional and Detailed Design Documents**
- **Software Integration and Test Plans**
- **Test and Verification Matrix**
- **Software User's Guide**
- **Release Definition Letters**
- **Delivery, Installation, Operations, and Maintenance Plan(s)**
- **Problem Tracking & Prioritized Work-off Schedule**

All Software will need maintenance

- **Maintenance must be included in planning**
- **What and how much depends on the use of the software**
- **Maintenance most often is done in response to existing known software errors, to add functionality, or to adapt to a changed environment**

GSFC Problem Summary Perspective

- Most mission systems include Mission Critical Software (MCS) elements for: Control Centers, S/C FSW, Instrument FSW, Planning & Scheduling, and Orbit/Attitude Navigation

- Over the last 5 years GSFC has held responsibility for well over 25 missions or well over 125 MCS elements

- GSFC has experienced 3 significant problems, computing to less than a 2.5% significant problem rate:
  - EOS FOS, EO-1 FSW, and IRAC FSW

While few in number, each was significant and drew a lot of attention

- No GSFC software problem has directly contributed to in-flight damage; to the contrary, software is routinely used to compensate for problems on-orbit

- To further improve performance and to reduce hero mode dependence, pragmatic improvement steps can be taken

MAP FSW - One of Many Successes

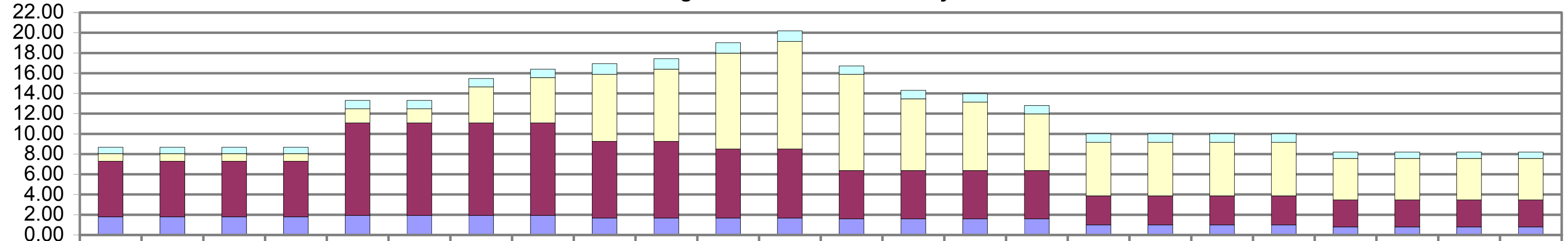
Date	Cost Relevance	Launch	Bodies (or MY)
Fall 1995	FSW Start	May 2001	realistic cost
1996 - 2000	Single string--> almost entirely fully redundant sensors/actuators/CPU		
	2 new testers added		realistic cost +2
Spring 2000	FSW Accept. Test/Red Team Review stunned at FSW readiness		
Winter 2000		December 2001	cost still in scope
December 2001	Launch		'virtually on-cost'

Some MAP FSW Points of Note

- Very FSW sensitive MAP Project Mgmt. and Systems Engineering
  - Supported honest cost estimates & supported FSW issues
- EO-1 Space Act Agreement meant MAP FSW was deliverable to EO-1 (earlier launch). Problems related to coordination of FSW deliveries & support to EO-1 were routine for 3 years.
- Reused XTE FSW plus XTE FSW staff -- rehosted to new CPU, memory,RTOS
- Single FSW Team managed by single FSW Sys. Mgr.
- FSW Team co-located with ground systems, GN&C analysts, ESC hardware developers.
- GN&C analysts & FOT were active members of the FSW Test Team.
- Developed C compiler for R000 CPU rather than using assembler.
- New FSW Executive for the R000 was a problem, but the team dealt with it.
- FSW for alternate star trackers became a requirement (==> 2 separate FSW loads).
- Outstanding Immediate Replacement for FSW Sys. Mgr. who resigned.
- Experienced and independent BT/Sys. Test Lead was involved from 'day 1'.
- 5 mos. delay in receipt of hardware for FSW testbed was mitigated by the team.
- Strong controls. Pretty good processes.
- Excellent FSW Requirements. Excellent FSW Test Traceability and Scenarios.

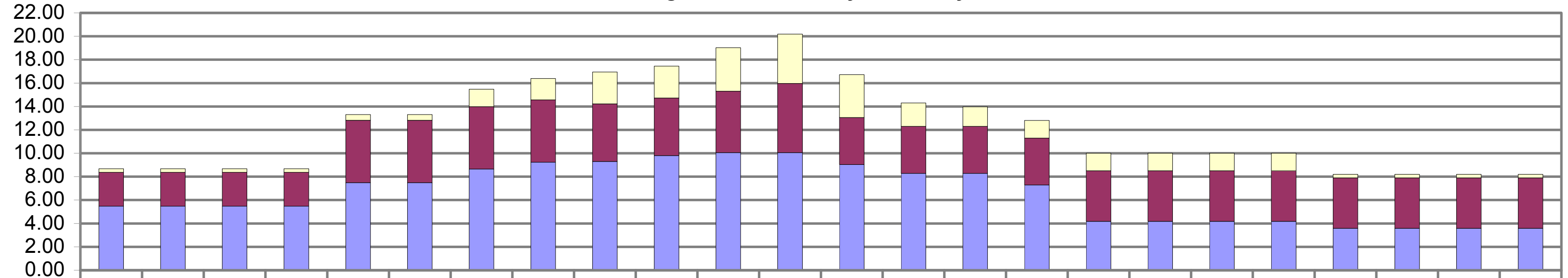
MAP FSW Staffing Profiles

Staffing Profile - Roles - Quarterly Actuals



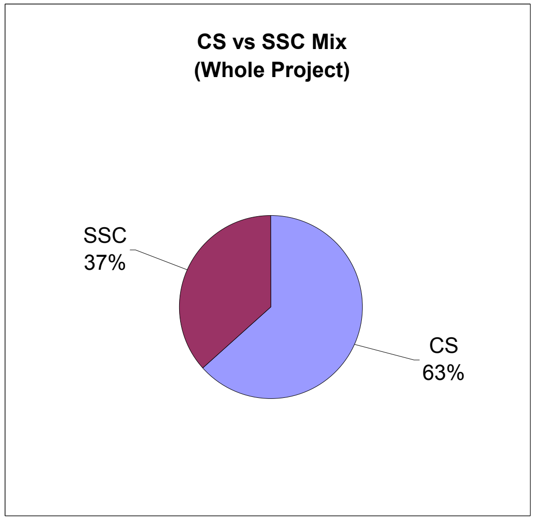
	OND 1995	JFM 1996	AMJ 1996	JAS 1996	OND 1996	JFM 1997	AMJ 1997	JAS 1997	OND 1997	JFM 1998	AMJ 1998	JAS 1998	OND 1998	JFM 1999	AMJ 1999	JAS 1999	OND 1999	JFM 2000	AMJ 2000	JAS 2000	OND 2000	JFM 2001	AMJ 2001	JAS 2001
□ Fac/Tools	0.63	0.63	0.63	0.63	0.83	0.83	0.83	0.83	1.05	1.05	1.05	1.05	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.63	0.63	0.63	0.63
□ Test	0.74	0.74	0.74	0.74	1.39	1.39	3.56	4.47	6.63	7.13	9.46	10.63	9.52	7.10	6.77	5.60	5.30	5.30	5.30	5.30	4.10	4.10	4.10	4.10
■ Dev	5.50	5.50	5.50	5.50	9.15	9.15	9.15	9.15	7.58	7.58	6.83	6.83	4.77	4.77	4.77	4.77	2.87	2.87	2.87	2.87	2.67	2.67	2.67	2.67
■ Eng/Mgmt	1.80	1.80	1.80	1.80	1.94	1.94	1.94	1.94	1.68	1.68	1.68	1.68	1.60	1.60	1.60	1.60	1.00	1.00	1.00	1.00	0.80	0.80	0.80	0.80

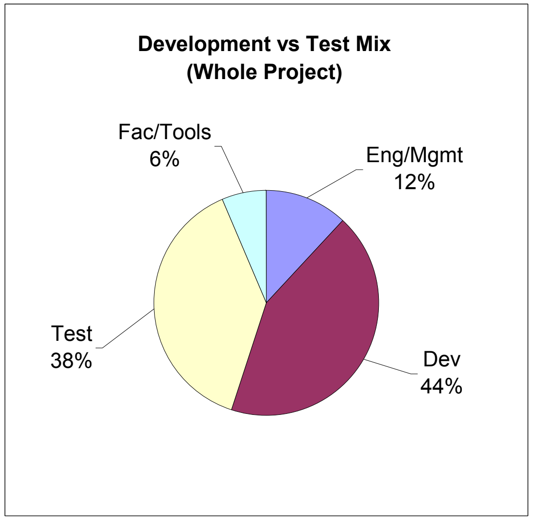
Staffing Profile - Seniority - Quarterly Actuals

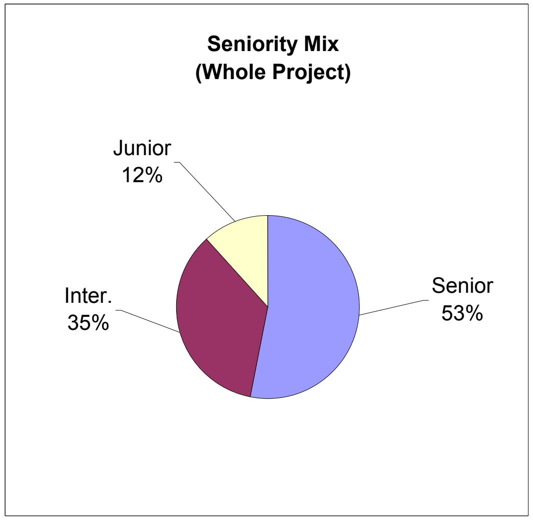


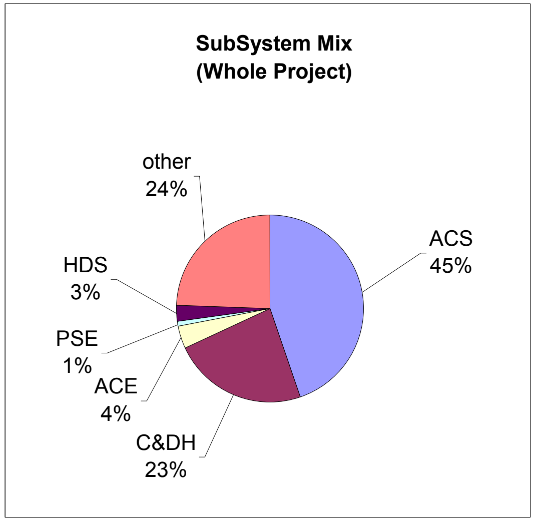
	OND 1995	JFM 1996	AMJ 1996	JAS 1996	OND 1996	JFM 1997	AMJ 1997	JAS 1997	OND 1997	JFM 1998	AMJ 1998	JAS 1998	OND 1998	JFM 1999	AMJ 1999	JAS 1999	OND 1999	JFM 2000	AMJ 2000	JAS 2000	OND 2000	JFM 2001	AMJ 2001	JAS 2001
□ Junior (J)	0.30	0.30	0.30	0.30	0.50	0.50	1.50	1.83	2.72	2.72	3.72	4.22	3.67	2.00	1.67	1.50	1.50	1.50	1.50	1.50	0.30	0.30	0.30	0.30
■ Interm. (I)	2.89	2.89	2.89	2.89	5.33	5.33	5.33	5.33	4.92	4.92	5.25	5.92	4.02	4.02	4.02	4.02	4.32	4.32	4.32	4.32	4.32	4.32	4.32	4.32
■ Senior (S)	5.48	5.48	5.48	5.48	7.48	7.48	8.65	9.23	9.30	9.80	10.05	10.05	9.03	8.28	8.28	7.28	4.18	4.18	4.18	4.18	3.58	3.58	3.58	3.58

MAP FSW Staffing View on a 75SYr Effort









All deliveries on schedule and with full capabilities, with core system completed a year before launch …..

1 of 122 success stories over the last 5 years

GSFC Mission Operations Software

General Mission Software Context

Mission Software Good Basic Practices

Recent GSFC Software Problems

Steps to Successful Software Products

Improvements In-Process/-Plan

Summary

Mission Critical Development Problems: FOS (1 of 3)

- Problem:
  - The Flight Operations Segment (FOS) was a new control center architecture designed to support the operations of the EOS Terra, Aqua, ICESat and Aura spacecraft. The FOS was developed by Lockheed Martin under subcontract to the Raytheon Corporation as part of the EOS Core System (ECS) Performance Based Contract (PBC).
  - In 1998, Raytheon issued a stop work order to Lockheed Martin due to a myriad of performance and functionality problems with FOS.
- Consequence:
  - The Terra launch was delayed for over a year due in large part to problems with FOS and the time necessary to develop a replacement control center system.

Mission Critical Development Problems: FOS (2 of 3)

- Causes:
  - The development of the FOS was tightly coupled with / driven by the development of a first of a kind earth science data processing system (SDPS) of unparalleled scope also being developed under the ECS PBC
    - The FOS and SDPS were forced to identical schedules of two drops each
    - Imposed the SDPS development methodology on the FOS
      - Contractor underestimated coding effort due to lack of practical C++ experience
      - As soon as developers had experience with C++, they left for higher paying jobs
    - Key FOS infrastructure components delegated to another contract element charged with developing common components for both FOS and SDPS
      - The true commonality between FOS and SDPS was minimal
      - The "common" development organization fell apart
  - FOS relied heavily on common systems and COTS
    - A common user interface across all FOS subsystems
    - Distributed Computer Env. never worked & there was no real need for Sybase
  - Wrong metrics were established and tracked for FOS
    - Code & unit test were tracked accurately, giving the early impression that FOS was progressing on schedule
    - There was inadequate emphasis on actual mission functionality
  - Requirements were very high-level, and all were treated equally
    - The requirements were at a high-level & not prioritized in order to give the PBC maximum flexibility in the development process.

Mission Critical Development Problems: FOS (3 of 3)

- Recovery:
  - The FOS was replaced with a system which came to be known as the EOS Mission Operations System (EMOS). EMOS is composed of one of the FOS subsystems (Mission Management) which was salvageable, a Raytheon (the prime ECS contractor) COTS (with special tailoring) product called Eclipse for real-time spacecraft command and control and an Integral Systems Inc. tailored product called ABE for trending and analysis.
  - EMOS was used to support Terra launch, has been modified to support the recent launch of Aqua, and is currently successfully supporting Terra and Aqua operations. Additional modifications are underway to support Aura launch and operations.
  - The responsibility for developing the control center for ICESat was given to LASP.

Mission Critical Development Problems: EO-1 (1 of 3)

- Problem:
  - The EO-1 flight software was a new implementation based on a MAP initial development drop. The EO-1 flight software was part of a Space Act Agreement with SWALES and Litton Industries. Cost and schedule pressures greatly constrained FSW development staffing and no HiFi simulation test environment was funded. The plan was to do the FSW test & debug as part of early I&T.
  - As I&T was about to commence lots of performance, functionality, and configuration management issues were identified. The Project manager invoked a call for hero mode support.

- Consequence:
  - A very senior FSW CS specialist was pulled from other important duties. Funds suddenly became available for staff and equipment. The FSW was resolved concurrent with other mission element completions.

Mission Critical Development Problems: EO-1 (2 of 3)

- Causes:
  - The EO-1 flight software was part of a cost cap forcing minimal staff and minimal schedule approaches.
  - System FSW leadership was absent and had no representation/voice at the EO-1 Project level.
  - Multiple FSW efforts (EO-1 has eleven CPUs) reporting to different hardware subsystem managers.
  - The MAP 'heritage' FSW, while started before EO-1 award, was actually scheduled for launch after EO-1 and IMAGE. The MAP FSW had no flight or even I&T burn-in history, although MAP had both XTE and TRMM flight heritage.
  - The EO-1 FSW testbed was of low fidelity in its C&DH and distributed architecture.
  - The basic plan of using I&T to debug the FSW without first executing a thorough HiFi test program indeed demonstrated itself as too risky.
  - Few processes were shared across CPUs.

Mission Critical Development Problems: EO-1 (3 of 3)

- Recovery:
  - A Sr. FSW Lead CS was pulled from other work and assigned full time. He was given full systems engineering authority over all FSW activities, across all CPUs.
  - The new FSW Lead reported directly to the Project Manager.
  - The Project funded & established the required HiFi testbed and funded additional FSW staff.
  - Good engineering practices were defined and enforced across all CPUs.
  - Thorough HiFi testing was completed and the FSW stabilized.
  - The EO-1 FSW met EO-1 launch and has performed extremely well on-orbit.

Mission Critical Development Problems: IRAC (1 of 3)

- **Problem:**
  - **The IRAC flight software is a new science instrument implementation responsible for operating a complex SIRTF mission IR camera with many mechanisms and data modes. As initial hardware & software integration started at GSFC, in preparations for IRAC test & delivery, a handful of troubling problems began to emerge.**
  - **The sole FSW person working this ‘ firmware’ effort took a job outside of NASA. Initial recovery efforts proved a discovery process, with new problems surfacing as fast as others were resolved.**

- **Consequence:**
  - **A very senior FSW CS specialist (having just worked EO-1) was pulled from other technical & management duties.
    Funds suddenly became available for staff and equipment.
    IRAC delivery to payload integration was delayed about a year.**

Mission Critical Development Problems: IRAC (2 of 3)

- Causes:
  - The IRAC FSW was declared ‘ firmware’ to hold costs
    - avoiding some perceived software process overhead
    - avoiding the need to address change development/verification once the PROMs were burned
  - The IRAC firmware/FSW had no representation/voice at the IRAC instrument level nor SIRTF mission level.
  - Do it all as a one person job working hand-in-glove with the electronics engineers. As IRAC I&T was gearing up the ‘ one person’ took another job outside of NASA.
  - The baseline testbed lacked the fidelity to test concurrent detector readouts, central to IRAC normal operations. Testing the FSW concurrent with electronics debug on the same platform was not viable.
  - Other NASA mission failures late in the IRAC development phase shifted the IRAC reference from one of success to one of near assured success. This drove IRAC into a requirements verification effort far beyond initial plans.

Mission Critical Development Problems: IRAC (3 of 3)

- **Recovery:**
  - A Sr. FSW Lead CS was pulled from other work and assigned full time. He was given full systems engineering authority over all IRAC FSW activities.
  - Good engineering practices were defined and enforced, ranging from establishing a Change Board to using earned value in measuring real progress.
  - The Project funded & provided the HiFi testbed needed. The Project also funded significant additional FSW staff (4 SSCs and 6 CSs).
  - A formal FSW requirements review (update) was conducted with SAO, JPL, and Lockheed. A design and code walktrough was conducted.
  - A thorough HiFi testing program keyed to documented requirements and cross-referenced to S/C level and instrument verification requirements was completed and the FSW stabilized.
  - The IRAC instrument with FSW has performed extremely well through Ball Aerospace instrument integration and cryo testing and into on-going S/C level testing at Lockheed.

Mission Critical Software Problems … Lessons Learned

- Software/Operations needs a voice at the Project mission level.
- Software / Operations knowledgeable senior managers, with (budget) authority and responsibility for the efforts, is a major key to success.
- Incremental development and integration keyed to important functionality and test of SW systems saves life cycle cost and reduces risk; however, it requires money and coordination up-front.
- High fidelity simulation systems are centrally important in test efforts.
- Test, test, and re-test, and "test as you fly" are keys to successful development.
- Many Project Managers have limited software experience & insight and press software schedule and staffing plans too hard to help meet competitive cost caps.
- Issues need to be driven up through both the Engineering and Projects management chains.
- With increased SW complexity (e.g., multiple onboard computers) and functionality, comes the need for an expanded Project system verification and validation program which cannot be compromised. Increased capability also means increased SW costs.

GSFC Mission Operations Software

General Mission Software Context

Mission Software Good Basic Practices

Recent GSFC Software Problems

Steps to Successful Software Products

Improvements In-Process/-Plan

Summary

Steps To Mission Critical Software Success… Development Org's Perspective

Follow Known Good Engineering Practices And Reject Pressures To The Contrary … Raise Issues Through Both Engineering and Project Senior Management Chains

- **Ensure dedicated & experienced software systems staff from concept through launch**
- **Get software presence at the Project Manager table**
- **Ensure thorough requirements definition and sign-off**
- **Resist program pressures toward schedule and staffing compressions and resist unrealistic budget-induced cuts (use data & experience with recent comparable systems)**
- **Get good people and delegate responsibility and authority to get things done**
  - **experienced team lead & key team member continuity are essential**
- **Measure progress against a known team schedule in some earned value process**
- **Conduct standard formal reviews (requirements, design, test, operations readiness)**
- **Conduct & document design and code walkthroughs, with customer & external experts**
- **Ensure high fidelity test environments with ETUs for flight SW**
- **Insist on a thorough software and mission test program. Test as you fly when you can**
- **Test failure and contingency scenarios and then test even more of them**
- **Involve the flight team in system and mission test definition and execution**
- **Establish and follow sound CM practices**

Steps to Mission Critical Software Success… Project Perspective

Acquire a team with good software engineering practices and provide the resources to get the job done. Make Sr. SW Managers part of the Project team. Conduct an extensive test program. Ensure SE & Ops participation.

- **Choose an experienced Development organization... good processes and proven people**
- **Provide adequate budget and schedule for proper software processes to be followed**
- **Acquire knowledgeable and experienced Senior Software Managers for Flight Software and Ground Systems & Operations -- who report directly to the Project Manager with budget and technical management authority**
- **Provide a strong test program lead manager advancing "test as you fly" (test beds & flight vehicle) and coordinating incremental delivery & test of flight and ground capabilities**
- **Provide systems engineering support for HW/SW trades and life cycle trades**
- **Insist upon and support design reviews, peer reviews, code walkthroughs, etc**
- **Establish and conduct an effective and tailored Project Verification & Validation effort**
- **Insist upon and support software PA and QA best practices**
- **Provide and follow a SW CM plan**
- **Ensure that the HW test engineers schedule time to support SW Project Verification & Validation and operations development**
- **Involve the Operations team in the SW Project Verification & Validation efforts**
- **Conduct extensive simulations and training exercises**
- **The use of a single Ground Data System & database for FSW test, S/C integration and test and operations reduces risk and lifecycle cost; however, it costs money up-front**
- **Retain adequate SW development maintenance staff into the operations phase**

Alarm Signs for Potential Problems (1 of 2)

Requirements Process:

Unbaselined requirements and interfaces.

Requirements instability (high rate of change reflecting immature definition).

Lack of requirements detail and/or requirements in discovery.

Operations Concept isn't matured for all mission life phases.

Build & Test Process:

Software processes aren't tailored based on mission experiences/risks.

Inadequate incremental delivery plan without real per build functionality.

Compressed schedules portrayed in few builds/releases and sacrifices in the test schedule (made to fit solutions).

Imposed software staff levels vs. well thought out approach (unusually low).

Lack of walkthroughs and peer reviews, with effective customer involvement.

Inadequate test plans and tools.

Poor fidelity simulation test environments.

Schedules seem to be constantly replanned with loss of the previous baseline.

Dependent deliveries from external organizations are delayed (e.g., ETU h/w, ICDs).

Poor CM.

Alarm Signs of Potential Problems (2 of 2)

<u>Systems Management:</u>

Software systems engineering isn't actively involved during early mission definition or design phases -- participate in trades, explain software impacts of decisions, ...

A single Software Manager is assigned as responsible for flight, ground and operations.

FSW Systems and Ground Systems/Operations Managers are not an integral part of the Project Manager's immediate team.

FSW Systems Manager isn't given budget or influence over all mission flight software.

Ground Systems & Ops Manager isn't given budget or influence over all mission ground and operations elements.

Experienced mission software engineering specialists are not in key lead positions.

Project level systems engineering isn't effective in mitigating software risks.

Systems engineering doesn't consider multiple solution sets or effect timely decisions.

Technical status reporting to Management isn't frequent or comprehensive.

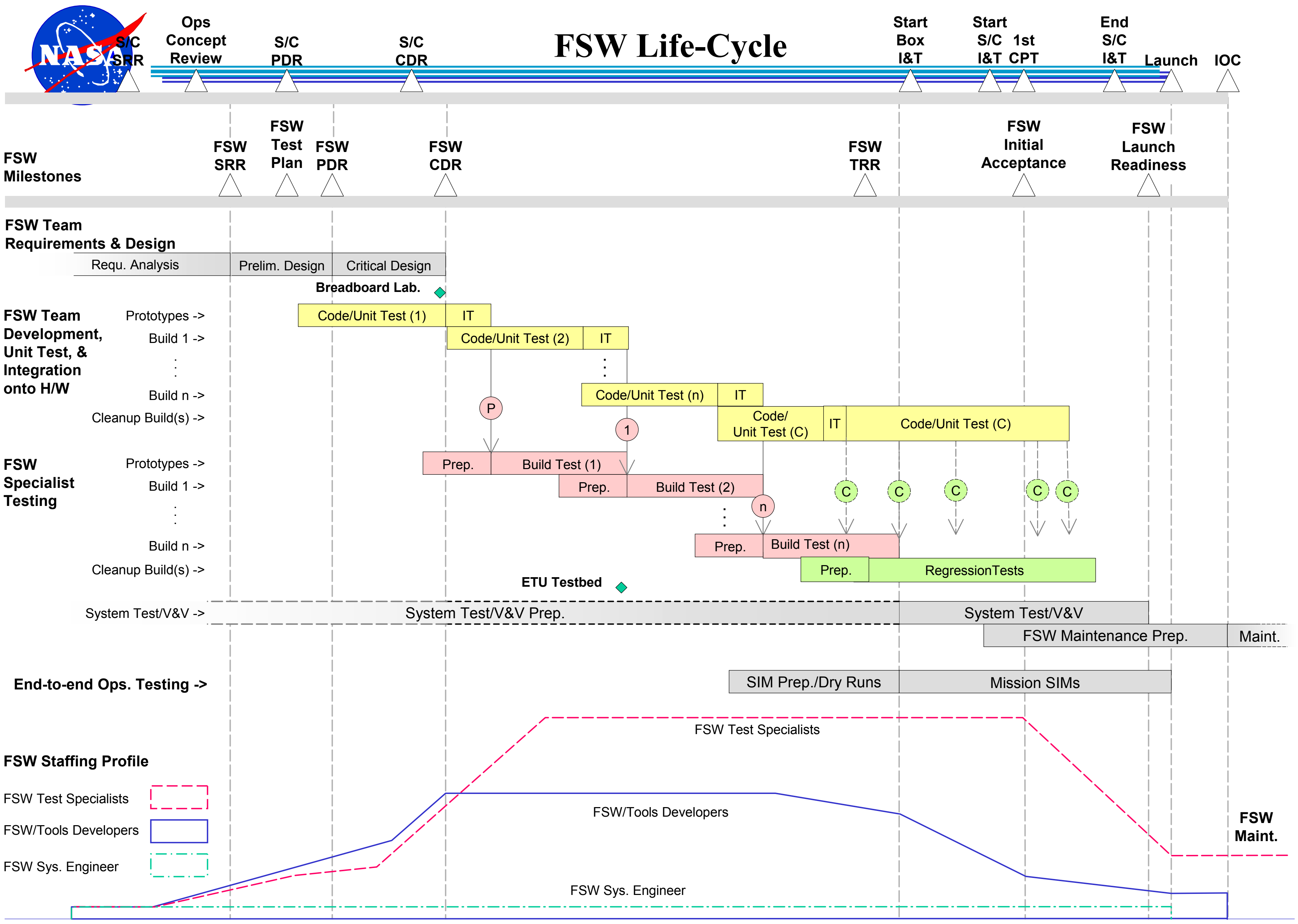
In-House Engineering's management involvement with the Project is distant.

What Is A Software Life-Cycle ?

- **Most projects use iterative life-cycles, but all show some form of "requirements-design-build-test."**
- **Most life-cycles are depicted as GANTT-type schedules.**
- **Software life cycles have intermediate reviews and products that provide completion and coordination points with other project elements or other sub-processes**
  - **Plan/Commitment, Software Requirements,**
    Software Design, Code Inspection, Test and Operational Readiness Reviews

- **Maintenance generally follows this same implementation cycle 'in the small'.**

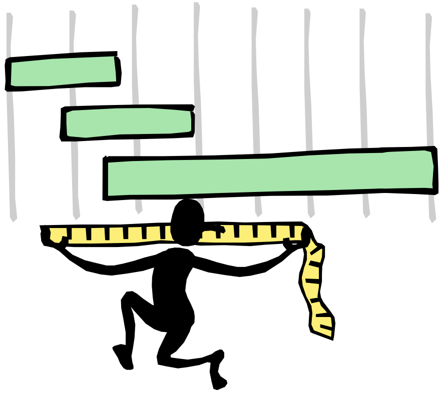
FSW Life-Cycle



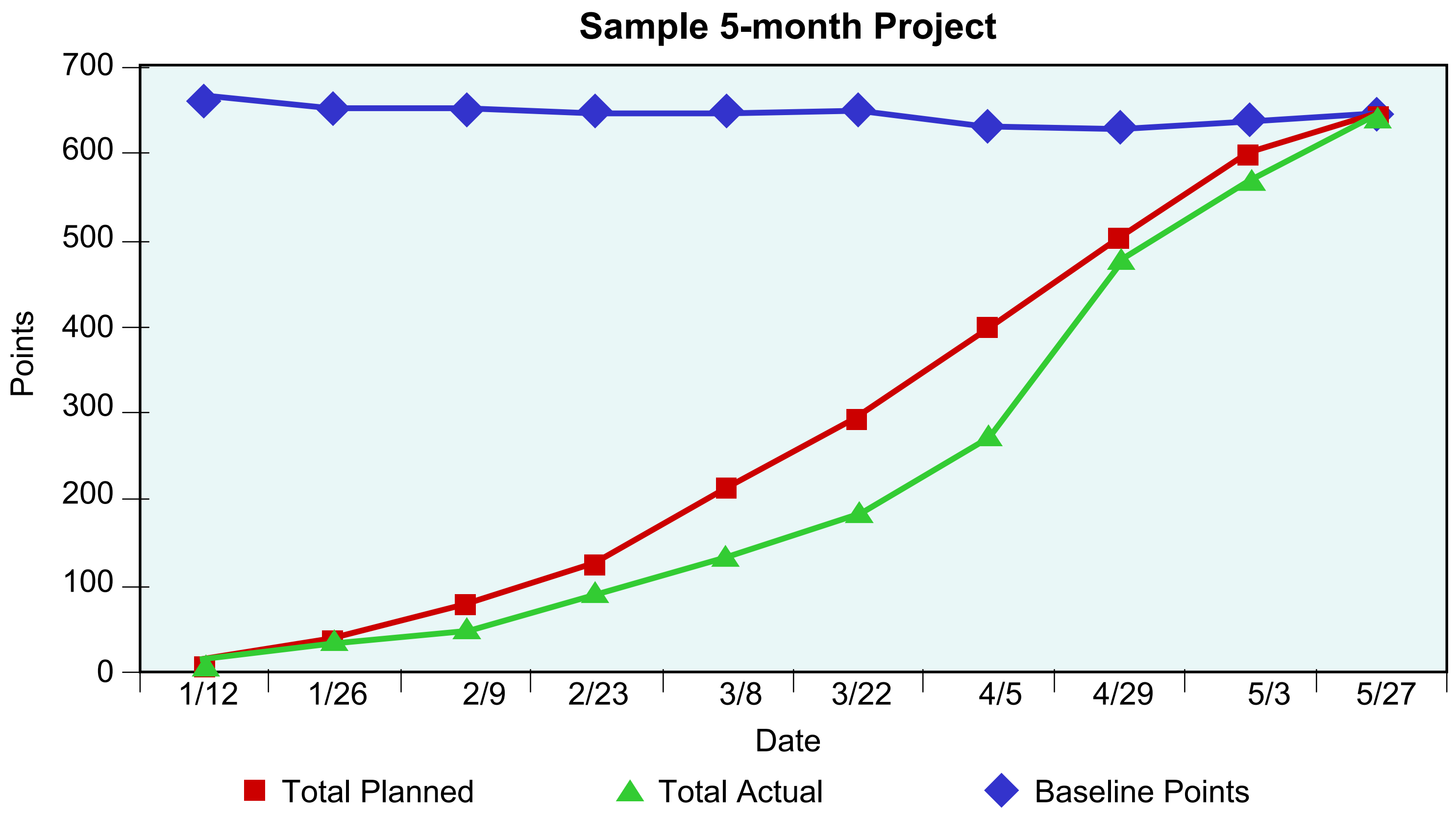
7/18/00

Measuring to Assist Project Management

- Measuring progress
  - Are the software development activities on schedule?
    - Earned Value Management (EVM) is our best and most successful measurement example.
    - Tie EVM to all life-cycle products,
      not lines of code or only the number of modules.
  - Should I make changes? What types of changes?
- Measuring quality
  - Will the software perform correctly?
  - Will the software fail? at critical times?
- Measuring functionality
  - Will the software do all the things it is expected to do?
  - Will all capabilities be included?



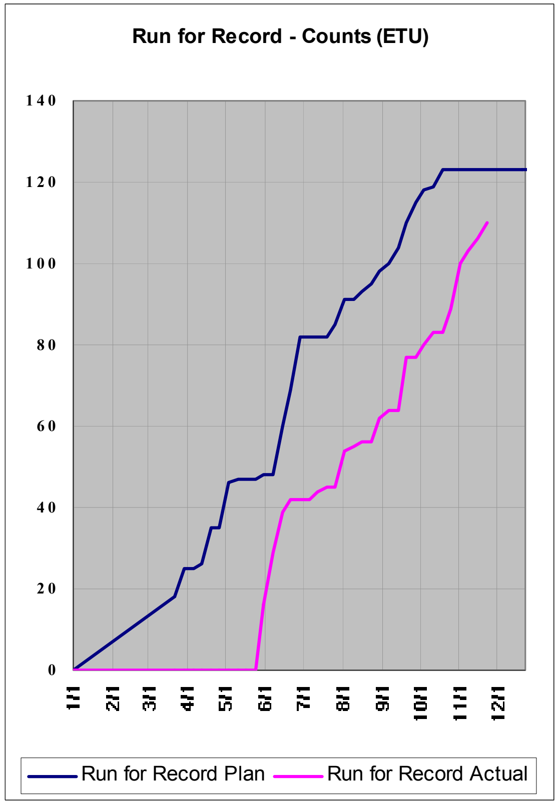
Simple Example of Managing Progress



Earned value is an excellent goal utilizing measures for management.

Each module (165) assigned 4 points: 1-designed; 2- coded; 3-inspected; 4-integrated

Sample Earned Value Engineering Build Test Profile



Current GSFC Mission Software Test Process (1 of 2)

Type	Purpose	Who
Unit Tests	Confirm code design and logic	Programmers
Integration Tests	Integrate Units together into target hardware environment	Development Team
Build Tests	Verify using high fidelity simulation environment that requirements were implemented correctly	S/W Test Specialists, Subsystem Experts,
*System Tests	Validate using high fidelity simulation environment that the defined software system supports exhaustive operations and contingency scenarios	S/W Test Specialists, maintenance staff, Subsystem Experts, Ops Team
**Acceptance Test	Execute the full set of systems tests on the 'final' release of the s/w	Same as systems

Notes: * Systems Test Readiness Review precedes start of systems tests.
** Software Acceptance Test Review follows completion of AT

Current GSFC Mission Software Test Process (2 of 2)

Type	Purpose	Who
HW-SW Integration Tests	Confirm h/w - s/w interfaces	H/W & S/W eng'rs
Regression Tests	Confirm that s/w changes don't impact previously working functionality and performance.	S/W Test Specialists
Stress Tests	Confirm that Maximum CPU, I/O, etc. loads don't impact performance.	S/W Test Specialists
End-to-end Tests	Validate all flight/ground data flow scenarios	Operations Staff, S/W Test Teams, Subsystem Experts
Mission Simulations	Exercise nominal and 'surprise'-anomalous operational scenarios (Launch thru science ops.)	Mission Systems Engineering, Operations Staff, Subsystem Experts, Launch support team

GSFC Mission Operations Software

General Mission Software Context

Mission Software Good Basic Practices

Recent GSFC Software Problems

Steps to Successful Software Products

Improvements In-Process/-Plan

Summary

Software Management Improvements In Process
- Organizational

- Established a prototype new business WBS template for in-house development including software as a direct Project level reporting element (vs. subsystem embedded).
- A senior flight software systems person and a senior ground systems & operations person should be part of the key Project Manager staff.
- Project level roles have been established on missions like SDO, GPM, and JWST.
- Over the last year the GSFC Applied Engineering & Technology (AET) and Flight Programs & Projects (FPP) Directorates have jointly reasserted a strong AETD organizational technical product responsibility. This is reflected in:
  - the establishment of per Project AETD Engineering Panels providing monthly AETD Management status briefs
  - the ISD has also established Branch bimonthly Technical Status briefs.
- AETD's recognition of software system significance is reflected in part in the selection of GSFC's current Chief Engineer (very experienced in Project software development)

Software Management Improvements In Process
- Technical

- Documented practices for software development across GSFC as part of ISO (a general resource). Need to be vigilant in assuring compliance to basic good engineering practices:
  - ISD Product Development Handbook (per product plan signed by Project & ISD)
  - GPG on Software Development and Maintenance
- A series of ' FSW Next ' discussions over the last six months , motivated in large part in response to Project cost concerns, identified the need for:
  - FSW Roadmap and evolution planning ...
  - Improving functional reuse using packaging concepts of domain modeling and information bus publishing/subscription systems with the potential for significant long term risk reduction and cost avoidance , but …

………… there is little to no funding or specific project interest.

- Active in evaluating the benefits/applicability of the CMMI continuous model to GSFC critical software products…

KEY CONCERN:

<u>Demonstrated 'value added' is essential to promote effective change</u>

July '02 MCS Colloquium Follow-Up Actions

1. Review JPL's methodology & processes in identifying JPL MCS issues and assess what makes sense for further GSFC software look-back. Do appropriate additional GSFC MCS evaluations (JPL also looked at cost growth).

2. Establish software management/oversight classroom and rotation training assignments for Systems Engineers and Project Managers (look to ISD's TMT pitch, the SEED program, evaluate JPL's project manager training materials).

3. Conduct joint AET and FPP Directorate level software reviews of specific project MCS plans to help assure that in-question mission baseline schedules and resources are reasonable (QLR Team concept).

4. Energize and continue in-process improvement efforts: (i) CMMI, (ii) documentation of existing good MCS practices, and (iii) creation of a MCS risk management plan.

5. Define important mission critical infrastructure needs & benefits and recommend improvements, with approaches for FSW reusable core capability and tools. Provide advocacy & additional resources to advance reuse, including process standards and improvements.

6. Assure FSW and Ground System & Operations Lead Software Managers are at the Project Managers table for future missions

7. Define methods to measure progress in the above actions.

8. Identify useful metrics for reporting MCS progress/problems to GSFC & HQ management. How do the improvement steps above reflect in these metrics ?

MCS Colloquium Actions Plans (1 of 2)

ACTION	PLAN	LEAD
1. Additional GSFC MCS look-back study	3 to 4 staff months (1/4 CS & 1 SSC)	S. Green/583 and J. Donohue/584
2. SEng & PMgr training	6 staff months (1/3 CS & 1 SSC)	S. Godfrey and <u>$125k CMMI</u>
3. Joint AETD & FPPD MCS feasibility checks	Prototype efforts in place	Steve Scott/500 & J. Hennessy
4. MCS good practices doc and deployment	Multi year funds request: (1 FSW CS w/BF and 2 SSC for ‘03 - ‘05 and 1 SSC for deployment in ‘04 and ‘05) - $600k, $800k, and $800k	E. Shell/582 & S. Godfrey/583 & J. Hennessy <u>$300k CMMI</u>

MCS Colloquium Actions Plans (2 of 2)

ACTION	PLAN	LEAD
5. FSW tools & methods in reuse, architectures,...	Funds request (1 FSW CS w/BF & 1.5 SSC) - $425 perFY)	E. Shell/582 & J. Hennessy
6. At PMgrs Table	Confirm in NB WBS and look at SDO/GPM/ JWST	M. Chu/580
7. Reporting Progress	Actions 1 to 6 & 8	M. Chu/580
8. MCS Metrics	Earned value based schedules & issues chart ? How do improvements reflect in these ? (1/4 CS and 1/2 SSC)	M. Chu/580 <u>$100k CMMI</u>

GSFC CMMI Benefits & First Year Efforts

- Anticipated CMMI Benefits
  - More consistent engineering & project management
  - Less “fire-fighting”
  - More cost/schedule predictability
  - Easier to bring new people up to speed
  - Increased productivity and improved quality
  - Reduced cycle time
- GSFC FY02 Goals (Phase 1-Assessment Phase):
  - to benchmark different pilot areas of GSFC
  - get better cost & effort estimate for doing CMMI
  - evaluate improvement approach
  - begin establishing infrastructure to support improvement
- Pilots completed in 10/02. Evaluation summary by 1/02/03
- Little process improvement planned in this pilot year
- Completed planned baselining; Established infrastructure for improvement
- GSFC flight software, flight projects/system engineering/acquisition, and ground software have been informally assessed using external experts

Strategies/Plans for FY03

Phase 2: Improvement Phase (4-5 year period)

- Focus more on improvements --Will work with projects/managers to choose areas where greatest benefit can be obtained
- Will use continuous model of CMMI- focus on areas where GSFC feels it needs to improve
- Initial primary improvement areas will be in flight software
  - Documentation of existing best practices (& suggested improvements)
  - Tools, checklist, templates to support consistent use of practices
  - Will continue activities started in FY02
- Using flight software practices as a basis, best practices will be documented for all of ISD
- Will work with systems engineering to pilot a small improvement
- Will begin to assess software acquisition processes to identify improvement opportunities
- Phase in improvements with projects in early stages
- Plan to choose one primary flight project to focus on

GSFC Mission Operations Software

General Mission Software Context

Mission Software Good Basic Practices

Recent GSFC Software Problems

Steps to Successful Software Products

Improvements In-Process/-Plan

Summary

Software Management Considerations (1 of 2)

- **1. ISD's software cost estimations generally prove to be accurate. Projects often exceed schedule/cost because projects constraints force unrealistic baselines.**
- **2. You need the right skill mix on the team and a capable, experienced Software Manager to oversee & guide it all.**
- **3. Make sure the person(s) responsible for overall flight software and ground systems & operations development and integration report directly to the project manager.**
- **4. Never identify a critical software task as a one-person effort. Always have a second person (at least) to share the knowledge and the interface responsibilities.**
- **5. When planning resources, use the rules of thumb that the test effort for ground software should be at least 30% of the total effort, and the test effort for flight software should be about 50% of the total effort. Remember that this staffing continues through integration and test.**

Software Management Considerations (2 of 2)

- 6. Baseline software requirements early, and then manage "scope creep". Ensure software requirements are based on an accurate operations concept. Designate one person responsible for the software requirements document and make sure that person is experienced in & knowledgeable of comparable software.
- 7. Don't cut corners in your management processes by calling software "firmware". It still requires the rigorous management processes of software development.
- 8. A good testbed is a requirement. Flight software testbeds must have all the Engineering Test Units and simulators for each and every external interface.
- 9. Software is required to validate your testbed. This software is needed very early in the project's life-cycle.
- 10. Distributed flight systems are expensive. You need a testbed of each processor, and then a combined testbed.
- 11. Problems uncovered in Integration and Test will often be "fixed" in the flight software, driving flight software to always come in just under the wire.

… and In Conclusion

- While there are no "silver bullets", there are proven techniques to manage software.

- Even heroes will fail with inadequate resources.

- Software Management needs to be approached as the technical discipline it is, not as a "mysterious art".

- While software development is not necessarily "harder" or more difficult than hardware development, it IS different and needs to be managed accordingly.
  - A good software management/product development plan and meaningful tracking of progress can go a long way towards mitigating risk and ensuring success.

BACKUPS

ISD Mission & Strategy

ISD Mission *(our core business … our fundamental purpose)*

To provide high value information systems products, services and expertise, and to advance information technologies, which are aligned with customer needs.

ISD Strategic Goals *(the "critical few" targets most important to move us toward our vision)*

1. **Deliver high value products and services that satisfy customer needs.**
2. **Advance leading-edge information systems technology.**
3. **Build a diverse, talented, innovative, energized, internationally recognized, workforce of employees and managers.**
4. **Establish open, flexible, collaborative relationships with customers and partners.**
5. **Build a cohesive ' no walls' organization with effective inter & intra Branch communication and collaboration.**

Information Systems Division (Code 580)

ISD provides information systems and systems components, and expertise in all phases of the science mission implementation process. To further support our customers and maintain our expertise we provide leadership and vision in identifying, developing and/or sponsoring advanced and emerging information systems technologies

Approximately 300 civil servants

Website: http://isd.gsfc.nasa.gov/

New Business AETD Work Breakdown Structure

PROJECT X:

- Management
- Science
- Mission Systems Engineering
  - SE Management
  - Systems Design
  - Technical Evaluation
  - Software SE
- Spacecraft
  - SE
  - Electrical Systems
  - Mechanical Systems
  - Guidance, Navigation & Control
  - ……
  - Flight Software
  - S/C Ground Software
- Payload
  - …..
  - Instrument Flight Software
  - Instrument Ground Software
- Ground Systems
- Operations

What is CMMI?

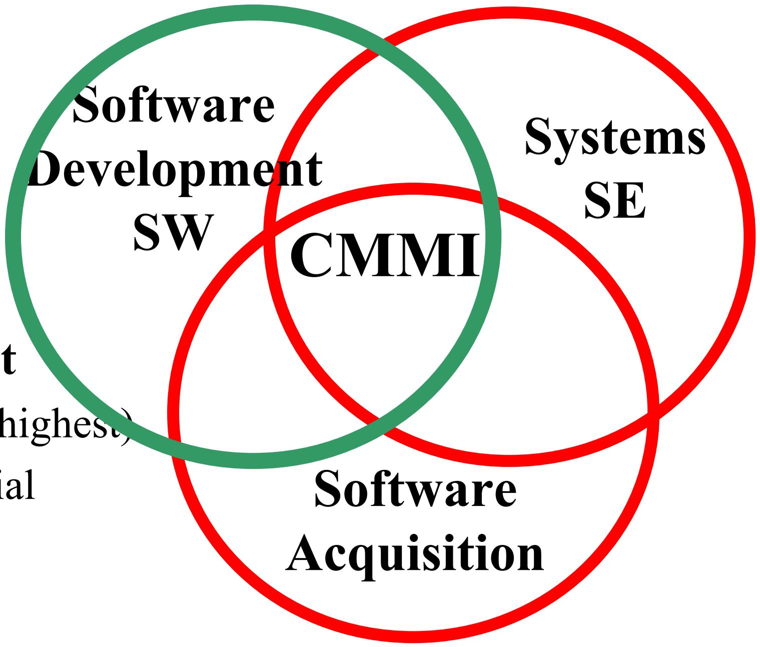
- The Capability Maturity Model Integrated (CMMI) is an **integrated framework for maturity models** and associated products that integrates the two key disciplines that are inseparable in a systems development activity: software engineering and systems engineering.

- A **common-sense application** of process management and quality improvement concepts to product development, maintenance and acquisition

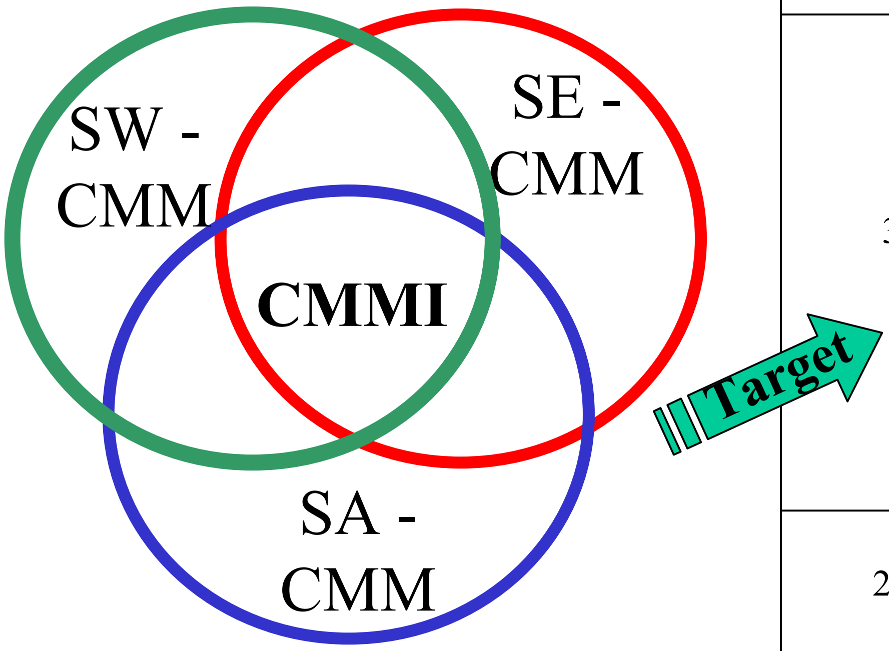
- A set of **best practices**

- A **community developed** guide

- A **model for organizational improvement**
  - CMMI divides capabilities into 5 levels (5 highest)
  - GSFC Goal of achieving level 3 as beneficial



Capability Maturity Model Integrated (CMMI)-Staged



Level	Process Areas
5 Optimizing	Organization innovation and deployment Causal analysis and resolution
4 Quantitatively Managed	Organizational process performance Quantitative project management
3 Defined	**Requirements development** **Technical solution** **Product integration** **Verification** **Validation** **Organizational process focus** **Organizational process definition** **Organizational training** **Integrated project management** **Risk management** **Decision analysis and resolution** **Integrated Supplier Management** **Integrated Teaming**
2 Managed	**Requirements management** **Project planning** **Project monitoring and control** **Configuration Management** **Supplier agreement management**
1 Initial	**Measurement and analysis** **Product & Process Quality Assurance**

CMMI and ISO

- ISO is a standard, CMMI is a model
- ISO is broad- focusing on more aspects of the business. Initially for manufacturing
- CMMI is "deep"- provides more in-depth guidance in more focused areas (Software/Systems Engineering/Software Acquisition-SW/SE/SA)
- Both tell you "what" to do, but not "how" to do it
- But CMMI tells you what "expected" practices are for a capable, mature organization
- CMMI provides much more detail for guidance than ISO by including an extensive set of "best practices", developed in collaboration with industry/gov/SEI

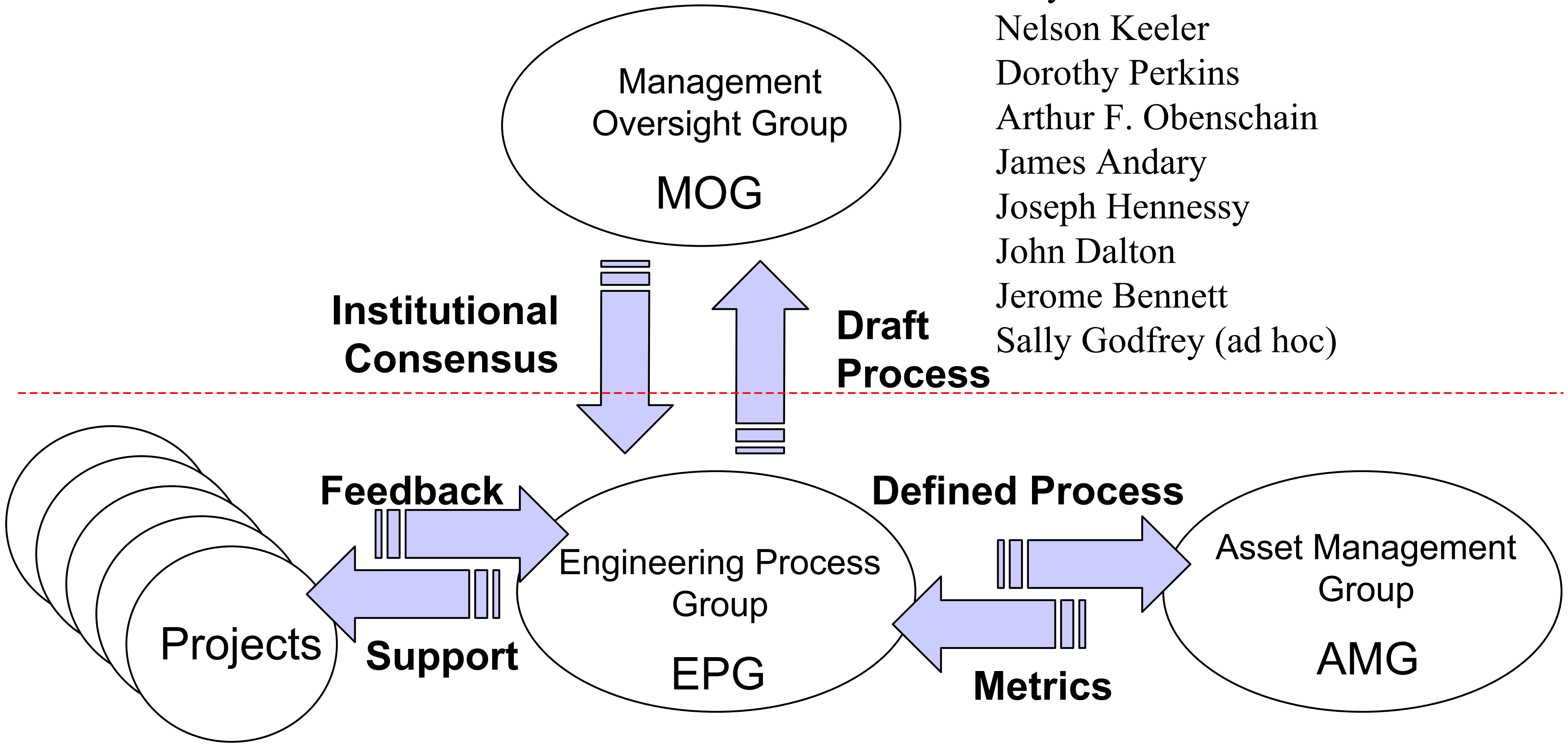
  -CMMI provides much better measure of quality of processes; ISO focuses more on having processes

  -CMMI puts more emphasis on continuous improvement

  -CMMI allows you to focus on one or a few process areas for improvement (It's a model, not a standard, like ISO) --Can rate just one area in CMMI

  -CMMI and ISO are not in conflict: ISO helps satisfy CMMI capabilities; CMMI more rigorous

CMMI GSFC Structure



Dr. Linda Rosenberg
Judy Bruner
Nelson Keeler
Dorothy Perkins
Arthur F. Obenschain
James Andary
Joseph Hennessy
John Dalton
Jerome Bennett
Sally Godfrey (ad hoc)

Sara (Sally) Godfrey
Representatives from 580, 530, 100, 200, 300, 400, 600

First Year CMMI FSW Area Progress

- Risk Management
  - Developed a prototype risk tracking system
  - Testing with data from two projects
  - Identifying lessons learned to feed into risk management process
- Cost Estimation
  - Generic cost estimation techniques examined
  - Developing material for a tutorial (presented at JPL at May'02 QMS WS)
  - Documenting flight software process
  - Working details with FSW people
- Review Guidelines
  - Drafted review guidelines
  - Incorporating flight software feedback
- Unit Test
  - Defined initial unit test standard; working to tailor for ST-5 use
  - Working to define metrics
- Inspections
  - Defined inspection process developed on JPL/GSFC research task
  - Working to tailor process & training initial FSW staff for first use